



## BASE DE DATOS

PROFESOR:

Ing. Yadira Franco R

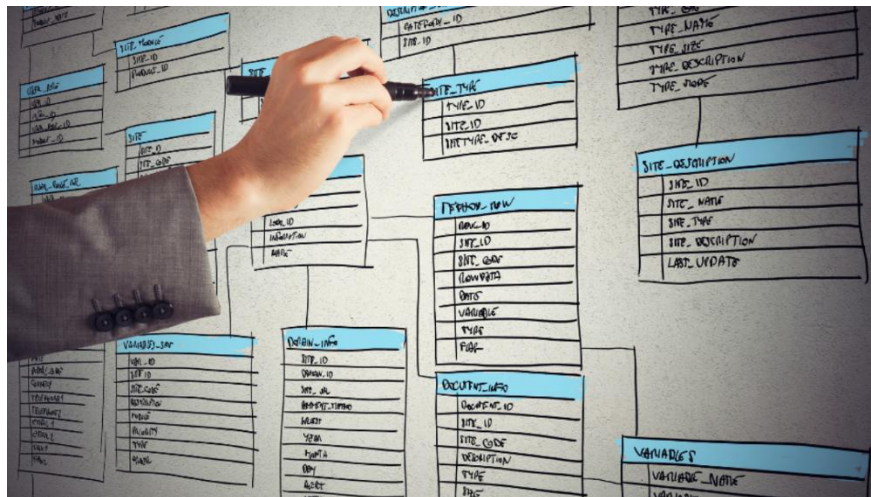
PERÍODO ACADÉMICO:

2024-B

## TAREA

TÍTULO:

## INVESTIGACIÓN Y PRACTICA



Estudiante

Paulo Cisneros

2024-B

## INVESTIGAR QUE SON Procedimientos Almacenados en Bases de Datos

- Entender qué son los procedimientos almacenados y cómo funcionan.
- Aprender a crear procedimientos almacenados sencillos.
- PRACTICA - Realizar operaciones de **INSERT**, **SELECT**, **DELETE** y **UPDATE** usando procedimientos almacenados.
- **Revisión de Buenas Prácticas**

## Introducción a los Procedimientos Almacenados **MSQL- PostgreSQL – Sql Server**

### 1. Concepto y Beneficios de los Procedimientos Almacenados

- **Explicación:** Los procedimientos almacenados son conjuntos de instrucciones SQL que se guardan y ejecutan en el servidor de base de datos. Permiten ejecutar operaciones complejas, con seguridad, rendimiento optimizado y reutilización de código.
- **Beneficios:**
  - Reutilización de código.
  - Mejora en la seguridad (al evitar inyecciones SQL).
  - Optimización en el rendimiento de consultas frecuentes.
  - Consistencia en las operaciones realizadas.

### 2. ESPECIFICAR LA Sintaxis Básica de un Procedimiento Almacenado

- **Explicación:** El delimitador se cambia temporalmente para permitir el uso de **;** dentro del procedimiento.

#### Crear la tabla de cliente:

```
CREATE TABLE cliente (  
    ClienteID INT AUTO_INCREMENT PRIMARY KEY, -- Campo para el ID único del cliente  
    Nombre VARCHAR(100), -- Campo para el nombre del cliente  
    Estatura DECIMAL(5,2), -- Campo para la estatura del cliente con dos decimales  
    FechaNacimiento DATE, -- Campo para la fecha de nacimiento del cliente  
    Sueldo DECIMAL(10,2) -- Campo para el sueldo del cliente con dos decimales  
);
```

```

6 • CREATE TABLE cliente (
7     ClienteID INT AUTO_INCREMENT PRIMARY KEY, -- Campo para el ID único del cliente
8     Nombre VARCHAR(100), -- Campo para el nombre del cliente
9     Estatura DECIMAL(5,2), -- Campo para la estatura del cliente con dos decimales
10    FechaNacimiento DATE, -- Campo para la fecha de nacimiento del cliente
11    Sueldo DECIMAL(10,2) -- Campo para el sueldo del cliente con dos decimales
12 );
13
14
15

```

### 3. Ejercicio 1: Crear un procedimiento simple que seleccione datos de la tabla cliente

```

14 -- Proceso para mostrar tabla cliente
15
16 delimiter //
17 create procedure mostrarCliente()
18 begin
19     select *from cliente
20 end;
21 delimiter //
22

```

En este código podemos ver que se le guarda en el procedimiento una consulta que nos mostrara todos los registros de la tabla cliente.

### 4. Ejercicio: Ejecutar - LLAMAR el procedimiento

```

30 -- Llamar procedimiento
31
32 • call mostrarCliente();
33

```

Result Grid

	ClienteID	Nombre	Estatura	FechaNacimiento	Sueldo
▶	1	Juan Pérez	1.75	1990-05-15	3000.00
	2	María Gómez	1.65	1985-10-20	2500.50
	3	Carlos López	1.80	1992-03-30	4000.00
	4	Ana Martínez	1.60	1988-07-25	2800.75
	5	Luis Fernández	1.70	1995-12-05	3200.00

Aquí llamamos al procedimiento y ejecuta la consulta que se escribió en el.

## Inserción, Actualización y Eliminación de Datos

### 1. Procedimiento de Inserción (INSERT)

- Crear un procedimiento que permita insertar un nuevo cliente en la tabla cliente

- Ejecutar - LLAMAR el procedimiento

```
-- Procedimiento de insertar

DELIMITER //

CREATE PROCEDURE InsertarCliente(
    IN p_Nombre VARCHAR(100),
    IN p_Estatura DECIMAL(5,2),
    IN p_FechaNacimiento DATE,
    IN p_Sueldo DECIMAL(10,2)
)
BEGIN
    INSERT INTO cliente (Nombre, Estatura, FechaNacimiento, Sueldo)
    VALUES (p_Nombre, p_Estatura, p_FechaNacimiento, p_Sueldo);
END //

DELIMITER ;
```

En este procedimiento vemos que utilizamos “in” para poder especificar los campos que vamos a llenar en la tabla luego escribimos la sintaxis que utilizamos para agregar registros a una tabla.

```
call InsertarCliente("Paulo Cisneros",1.76,"2002-05-05",3500.00)
```

Result Grid					
Filter Rows:					
Edit: Export/					
	ClienteID	Nombre	Estatura	FechaNacimiento	Sueldo
▶	1	Juan Pérez	1.75	1990-05-15	3000.00
	2	María Gómez	1.65	1985-10-20	2500.50
	3	Carlos López	1.80	1992-03-30	4000.00
	4	Ana Martínez	1.60	1988-07-25	2800.75
	5	Luis Fernández	1.70	1995-12-05	3200.00
	6	Paulo Cisneros	1.76	2002-05-05	3500.00
*	NULL	NULL	NULL	NULL	NULL

## 2. Procedimiento de Actualización (UPDATE)

Actualizar la edad de un cliente específico:

```
-- Actualizar la edad

DELIMITER //

CREATE PROCEDURE ActualizarEdadCliente(
    IN p_ClienteID INT,
    IN p_NuevaEdad INT
)
BEGIN
    DECLARE nueva_FechaNacimiento DATE;

    -- Calcular la nueva fecha de nacimiento en función de la edad
    SET nueva_FechaNacimiento = DATE_SUB(CURDATE(), INTERVAL p_NuevaEdad YEAR);

    -- Actualizar el registro del cliente
    UPDATE cliente
    SET FechaNacimiento = nueva_FechaNacimiento
    WHERE ClienteID = p_ClienteID;
END //
```

```
call ActualizarEdadCliente(1,30);
```

Result Grid					
		Filter Rows:		Edit:	
	ClienteID	Nombre	Estatura	FechaNacimiento	Sueldo
▶	1	Juan Pérez	1.75	1994-12-17	3000.00
	2	María Gómez	1.65	1985-10-20	2500.50

## 3. Procedimiento de Eliminación (DELETE)

Eliminar un cliente de la base de datos usando su ClienteID:


```
-- eliminar cliente

DELIMITER //

CREATE PROCEDURE EliminarCliente(
    IN p_ClienteID INT
)
BEGIN
    DELETE FROM cliente
    WHERE ClienteID = p_ClienteID;
END //

DELIMITER ;
```

```
call EliminarCliente(1);
```

Result Grid					
Filter Rows: <input type="text"/>					
Edit: 					
	ClienteID	Nombre	Estatura	FechaNacimiento	Sueldo
▶	2	María Gómez	1.65	1985-10-20	2500.50
	3	Carlos López	1.80	1992-03-30	4000.00
	4	Ana Martínez	1.60	1988-07-25	2800.75
	5	Luis Fernández	1.70	1995-12-05	3200.00
●	NULL	NULL	NULL	NULL	NULL

## Introducción a Condiciones en Procedimientos Almacenados

### Uso de Condicionales (IF)

El uso de condicionales dentro de los procedimientos es fundamental para tomar decisiones basadas en los datos.

Verifica si la edad de un cliente es mayor o igual a 22:

```
CREATE PROCEDURE VerificarEdadCliente(  
    IN p_ClienteID INT  
)  
BEGIN  
    DECLARE edad INT;  
    DECLARE fechaNacimiento DATE;  
    SELECT FechaNacimiento INTO fechaNacimiento  
    FROM cliente  
    WHERE ClienteID = p_ClienteID;  
  
    -- Calcular la edad del cliente  
    SET edad = TIMESTAMPDIFF(YEAR, fechaNacimiento, CURDATE());  
  
    -- Verificar si la edad es mayor o igual a 22  
    IF edad >= 22 THEN  
        SELECT CONCAT('El cliente con ID ', p_ClienteID, ' tiene ', edad, ' años y es mayor o igual a 22 años.') AS Mensaje;  
    ELSE  
        SELECT CONCAT('El cliente con ID ', p_ClienteID, ' tiene ', edad, ' años y es menor de 22 años.') AS Mensaje;  
    END IF;  
END;
```

```
call VerificarEdadCliente(2);
```

## Creación de la Tabla de Órdenes CON RELACIÓN CON EL CLIENTE – FORANEA

```
-- Crear tabla de ordenes
• create table ordenes (
    OrdenID int primary key auto_increment,
    ClienteID int,
    Monto decimal (10,2),
    foreign key (ClienteID) references cliente(ClienteID)
);
```

---

Para almacenar las órdenes de los clientes, se debe crear la tabla **ordenes**:

- Procedimientos de Órdenes -Insertar Orden

```
-- Procedimiento de inserción
DELIMITER //

CREATE PROCEDURE InsertarOrden(
    IN p_ClienteID INT,
    IN p_Monto DECIMAL(10,2)
)
BEGIN
    INSERT INTO ordenes (ClienteID, Monto)
    VALUES (p_ClienteID, p_Monto);
END //
```

- Procedimientos Actualizar Orden

```
-- Procedimiento de Actualizar
DELIMITER //

CREATE PROCEDURE ActualizarOrden(
    IN p_OrdenID INT,
    IN p_NuevoMonto DECIMAL(10,2)
)
BEGIN
    UPDATE ordenes
    SET Monto = p_NuevoMonto
    WHERE OrdenID = p_OrdenID;
END //

DELIMITER ;
```

- Procedimientos Eliminar Orden

```
-- Procedimiento de eliminación

DELIMITER //

CREATE PROCEDURE EliminarOrden(
    IN p_OrdenID INT
)
BEGIN
    DELETE FROM ordenes
    WHERE OrdenID = p_OrdenID;
END //

DELIMITER ;
```

## Entrega Final

### Instrucciones de Entrega:

#### 1. Objetivos:

Crear procedimientos almacenados para **insertar, actualizar, eliminar y consultar** registros en las tablas cliente y ordenes.

#### 2. Archivo de Script:

Los estudiantes deben escribir y guardar el código SQL con todos los procedimientos mencionados.

#### 3. Documento PDF:

Incluir las capturas de pantalla y explicaciones detalladas de los pasos realizados durante la tarea.

#### 4. Subida a GitHub:

Subir el script .sql y el documento PDF a un repositorio en GitHub para su REVISIÓN